Démonstration d’asservissement visuel sur Pepper depuis ROS

# Prérequis :

Tutoriels de configuration.

Travail de lagadic

# But :

Effectuer un asservissement visuel SANS ROTATION :

* Sans mouvement entre l’image désirée et la courante
* Avec un léger déplacement entre l’image désirée et la courante

# Problèmes :

* Le streaming vidéo n’est pas fluide et Pepper change d’adresse ip à chaque démarrage.
* Pepper a un fonctionnement primitif le poussant à s’arrêter lorsqu’il croise un obstacle, mais il effectue aussi des mouvements incontrôlables de ses bras et de la tête lors de l’exécution d’un ordre depuis le topic /cmd\_vel.

# Solutions :

* Pour la partie connexion, il est nécessaire d’enregistrer l’adresse MAC de Pepper dans le réseau et de vérifier que le switch ne filtre pas les données en transit, la communication de Pepper est lourde de paquets (pour le streaming vidéo notamment).
* Effectuer des contre mouvements (ceux-ci sont implémentés dans le code de l’asservissement visuel sans rotation).

# Résultats et interprétations :

[Video](VisualServoing.mp4)

* Sans contre mouvements le corps de Pepper bouge (bras, tête) sans tendance particulière.
* Avec et avec une vélocité faible (0.01) sur les articulations, le streaming vidéo est fluide sans saccade dû à des mouvements brutal.

Il est nécessaire d’adapter le nœud visual\_servoing à Pepper, λ est un multiplicateur de vélocité, en simulation ordinateur il peut être égal à 0.7 mais pour utiliser un robot réel, 2.3 cela semble être un bon compromis entre précision et vitesse. Pepper pèse 30 kg, ses moteurs ont besoins d’un élan pour atteindre une inertie suffisante pour appliquer des petites valeurs de vélocité. Il peut être intéressant d’ajouter une première valeur plus élevée que les suivantes afin de donner aux moteurs cette inertie. On pourra alors réduire λ afin d’avoir une meilleure précision.

Néanmoins, l’asservissement fonctionne bien (de façon attendu) sans mouvement entre les deux images, mais aussi lors d’un léger déplacement (50cm max) sur l’axe x.

En effet il semble y avoir une tendance à mieux fonctionner sur un déplacement sur x plutôt que sur y. L’explication est que Pepper n’a que deux degrés de liberté dans ce nœud, alors la matrice d’interaction ne permet pas un asservissement efficace sur l’axe y.

# Mise en marche :

* Lancement du nœud desired\_image (prend la photo désirée) :

roslaunch rosovbc desired\_image\_manager.launch

* Lancement du nœud visual\_servoing (asservissement visuel, 3 fenêtres) :

roslaunch rosovbc visual\_servoing.launch

* Arrêt du mouvement /cmd\_vel :

rostopic pub -r 1 /cmd\_vel geometry\_msgs/Twist '{linear: {x: y: 0.0, z: 0.0}, angular: {x: 0.0,y: 0.0,z: 0.0}}'

* Contre mouvement manuel en boucle (10 fois par seconde) de vélocité faible (0.01) :

rostopic pub -r 1 /joint\_angles naoqi\_bridge\_msgs/JointAnglesWhSpeed '{header: auto, joint\_names: ["HeadPitch","HeadYaw","RShoulderPitch","RShoulderRoll","RElbowYaw","RElbowRoll","RWristYaw","RHand","LShoulderPitch","LShoulderRoll","LElbowYaw","LElbowRoll","LWristYaw","LHand"], joint\_angles: [-0.60,0,1.75,0,1.5,0,0,0.75,1.75,0,-1.5,0,0,0.75], speed: 0.01}'

* Valeurs des articulations pour la position repos :
  + HeadPitch -> -0.60
  + HeadYaw -> 0
  + RShoulderPitch, LShoulderPitch -> 1.75
  + RShoulderRoll, LShoulderRoll -> 0
  + RElbowYaw -> 1.5, LElbowYaw -> -1.5
  + RElbowRoll, LElbowRoll -> 0
  + RWristYaw, LWristYaw -> 0
  + RHand, LHand -> 0.75
* Déplacer Pepper avec le clavier (nécessite le package teleop\_twist\_keyboard) :

rosrun teleop\_twist\_keyboard teleop\_twist\_keyboard.py